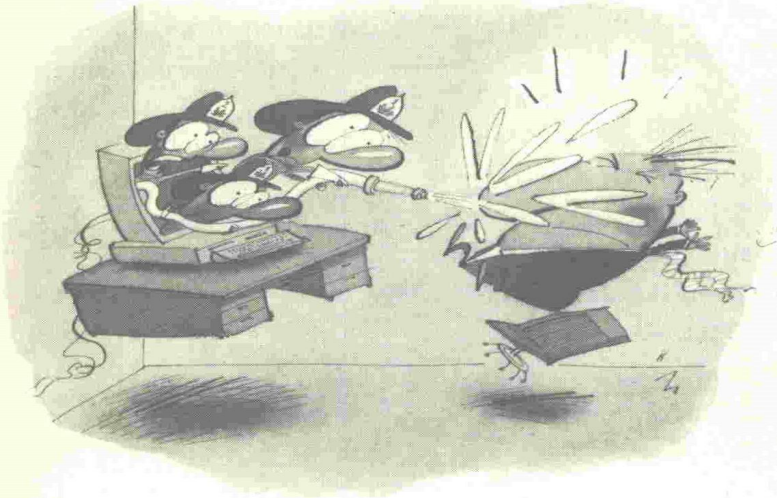
สุทธิศักดิ์ พงศ์ธนาพาณิช

Programmers's Notes for VB/Win ตอนที่ 2



```
Declare Function GetSystemDirectory Lib
"Kernel" (ByVal lpBuffer As String,
ByVal nSize As Integer) As Integer

Declare Function GetWindowsDirectory Lib
"Kernel" (ByVal lpBuffer As String,
ByVal nSize As Integer) As Integer

Sub Form_Activate ()
Dim UsedB%, Buffer As String * 256

UsedB% = GetSystemDirectory(Buffer,

Print "System :"; Left$(Buffer, UsdB%)

UsedB% = GetWindowsDirectory
(Buffer, 255)
Print "Windows:";Left$(Buffer, UsedB%)
End Sub
```

รวบรวมเทคนิคสำหรับผู้อ่านที่สนใจในการเขียนโปรแกรมด้วย VB/Win สามารถนำไปประยุกต์ใช้ได้อย่างรวดเร็ว

ในฉบับนี้ยังคงเป็นการรวบรวมเทคนิคพิเศษและคำถามจากผู้อ่านที่จดหมายมายังตู้ ป.ณ. ของผู้เขียน สำหรับผู้อ่านที่สนใจในการเขียนโปรแกรมด้วย VB/Win สามารถนำไปประยุกต์ใช้ได้อย่างรวดเร็ว

การค้นหาไดเรคทอรีของวินโดวส์

ในการเขียนโปรแกรมที่รันภายใต้วินโดวส์ การค้นหาไดเรคทอรีของวินโดวส์และระบบเป็นสิ่งที่จำเป็นอย่างยิ่ง เพราะโดยส่วนใหญ่ไฟล์ .INI มักจะติดตั้งในไดเรคทอรีของวินโดวส์และไฟล์ฟอนต์, ไฟล์ .DLL, .VBX, .OCX ก็มักจะติดตั้งในไดเรคทอรีของระบบ ซึ่งผู้อ่านสามารถค้นหาไดเรคทอรีดังกล่าวได้โดยใช้ฟังก์ชันวินโดวส์ API ที่ชื่อ GetWindowsDirectory และ GetSystemDirectory ตามลำดับดังนี้

บัญญัติ 10 ประการของการใช้ API

การเรียกใช้ฟังก์ชันวินโดวส์ API ด้วย VB/Win เป็นงานอย่างหนึ่งที่มีความเสี่ยงสูง เพราะว่าพารามิเตอร์ของฟังก์ชันวินโดวส์ API ส่วนใหญ่ไม่คอมแพตทิเบิลกับ VB/Win ดังนั้นก่อนที่ผู้อ่านจะทดสอบโปรแกรมควรคำนึงถึงหลักการณ์ดังต่อไปนี้

1. จัดเก็บโค้ดก่อนการทดสอบทุกครั้ง

2. ตรวจสอบพารามิเตอร์ที่ยังไม่มีคำสงวน ByVal นำหน้าโดยเฉพาะพารามิเตอร์ชนิดสตริง

3. หลีกเลี่ยงการใช้การประกาศประเภทข้อมูล As Any เพราะการส่งชนิดของข้อมูลผิดพลาดในกรณีนี้มักจะเกิดข้อผิดพลาด GPF ทันที

4. ข้อมูลชนิดสตริงของ VB/Win ไม่สามารถใช้กำหนดให้กับพารามิเตอร์ของฟังก์ชันวินโดวส์ API ได้เลย ซึ่งโดยส่วนใหญ่จะต้องแปลงให้อยู่ในรูปแบบภาษาซี (null-terminate string) โดยใช้คำสงวน ByVal นำหน้าเสมอ

5. ใช้ประโยค Option Explicit ในทุก ๆ โมดูล เพื่อป้องกันการใช้ตัวแปรที่ไม่เหมาะสม โดยเฉพาะตัวแปรชนิด variant

6. กรณีที่พารามิเตอร์เป็นสตริง ขนาดของสตริงที่กำหนดให้กับตัวแปรดังกล่าวต้องมากกว่าขนาดสูงสุดที่ฟังก์ชันต้องการใช้อยู่ 1 ไบต์เสมอสำหรับแอสกี 0 ที่ต่อท้ายสตริง

7. ตรวจสอบการประกาศฟังก์ชันและประเภทของตัวแปรจากหนังสือ Microsoft Windows Developmet Ket เพื่อความปลอดภัย

8. ถ้าหากเกิด GPF ในขณะทดสอบโปรแกรม ผู้อ่านควรจะบูตวินโดวส์ใหม่อีกครั้ง

9. ใช้โปรแกรม WPS.EXE ตรวจสอบการโหลดโมดูลของวินโดวส์ (ติดมากับ VB/Win 3.0)

10. ในกรณีที่ VB/Win มีคำสั่งที่เทียบเท่ากับฟังก์ชันวินโดวส์ API ก็ให้หันมาใช้คำสั่งดังกล่าวแทน เช่น คำสั่ง GetSetting ของ VB/Win 4.0 แทนฟังก์ชัน GetProfileString เป็นต้น

กำหนดพื้นที่ใช้งานของผู้ใช้

ผู้อ่านสามารถกำหนดพื้นที่สำหรับการเคลื่อนที่ของเม้าส์ภายใต้คอนโทรลต่าง ๆ เช่น command button, picture box หรือฟอร์ม เพื่อกำหนดพื้นที่การใช้งานของผู้ใช้ได้โดยการใช้ฟังก์ชันวินโดวส์ API ที่ชื่อ ClipCursor ดังนี้

```
Type RECT
left As Integer
top As Integer
right As Integer
bottom As Integer
End Type

Declare Sub GetWindowRect Lib "User" (ByVal
hWnd As Integer, lpRect As Any)
Declare Sub ClipCursor Lib "User" (lpRect As Any)

Sub Form_Activate ()
Dim Box As RECT

GetWindowRect Me.hWnd, Box
ClipCursor Box
End Sub

Sub Form_DblClick ()
ClipCursor ByVal 0&
End Sub
```

โดยเมื่อผู้อ่านรันตัวอย่างเม้าส์จะถูกจำกัดขอบเขตการเคลื่อนที่เฉพาะในฟอร์มเท่านั้น และเมื่อผู้อ่านดับเบิ้ลคลิ๊กภายในฟอร์ม ขอบเขตการเคลื่อนที่ของเม้าส์ที่ถูกจำกัดก็จะถูกยกเลิกทันที

การกำหนด read-only Text box

คอนโทรล text box มักถูกใช้ในการรับหรือแสดงผลข้อความ ซึ่งในบางครั้งถ้าหากเราต้องการใช้แสดงผลข้อความอย่างเดียว และไม่ต้องการให้มีการแก้ไขข้อความใน text box ก็สามารถกระทำได้โดยใช้ฟังก์ชันวินโดวส์ API ที่ชื่อ SendMessage ดังนี้

```
Declare Function SendMessage Lib "User"
 (ByVal hWnd As Integer, ByVal wMsg As
 Integer, ByVal wParam As Integer,
 lParam As Any) As Long

Const WM_USER = &H400
Const EM_SETREADONLY = (WM_USER + 31)

Sub Form_Click ()
Static ROflag%
 Dim Dummy&

 If ROflag% Then
   ROflag% = 0     ' Read/Write
 Else
   ROflag% = 1     ' Read only
 End If
 Dummy& =   SendMessage(Text1.hWnd,
            EM_SETREADONLY, ROflag%, 0)
End Sub
```

โดยกำหนดให้มี text box ที่ชื่อ Text1 ในฟอร์ม เมื่อรันโปรแกรม ผู้อ่านสามารถกำหนดให้ Text1 อยู่ในสถานะ Read only หรือ Read/Write ได้โดยการคลิ๊กในพื้นที่ของฟอร์ม

อ่านเวอร์ชันของวินโดวส์และดอส

ผู้อ่านสามารถอ่านเวอร์ชันของวินโดวส์และดอสภายใต้ VB/Win ได้ด้วยฟังก์ชันวินโดวส์ API ที่ชื่อ GetVersion ได้ดังนี้

```
Declare Function GetVersion Lib "Kernel" ()
        As Long

Function GetDosVersion () As Single
   Dim VerWord As Long

   VerWord = CInt(GetVersion&() \ &H10000)
   GetDosVersion = CSng(CInt(VerWord /256)
                & "." & (VerWord And &HFF))
End Function

Function GeWindowVersion () As Single
   Dim VerWord As Long

   VerWord = CInt(GetVersion&()
             And &HFFFF&)
   GeWindowVersion = CSng((VerWord And
     &HFF) & "." & CInt(VerWord / 256))
End Function
```

การย้ายคอนโทรลระหว่างฟอร์ม

ในการเขียนโปรแกรม ผู้เขียนมักจะใช้วิธีการย้ายคอนโทรลข้ามฟอร์มโดยเฉพาะ command button ค่อนข้างบ่อย เนื่องจากโปรแกรมที่ได้จะมีขนาดเล็ก และในบางกรณีโปรแกรมสามารถทำงานได้เร็วขึ้นด้วย ซึ่งเราจะใช้ฟังก์ชันวินโดวส์ API ที่ชื่อ SetParent ดังนี้

```
Declare Function SetParent Lib "User"
  (ByVal hWndChild As Integer, ByVal
   hWndNewParent As Integer) As Integer
```

เช่น สมมติว่าผู้อ่านสร้าง Command1 ไว้ในฟอร์ม Form1 และต้องการย้าย Command1 มายังฟอร์ม Form2 ก็สามารถกระทำได้โดยอาศัยโค้ดดังนี้

```
OldhWnd% = SetParent(Command1.hWnd, Form2.hWnd)
```

โดยที่ค่าที่ได้จากฟังก์ชันนี้ก็จะเป็นหมายเลข handle ของฟอร์มแม่ของคอนโทรล Command1 ก่อนย้ายหรือในที่นี้หมายถึง Form1 นั่นเอง

การแสดงเมนูชิดขอบขวา

ผู้อ่านที่เคยใช้ QuickBASIC ก็จะเห็นว่าเมนู Help นั่นอยู่ชิดขอบขวา ซึ่งในการสร้างเมนูด้วย Menu Design นั้นไม่สามารถกระทำได้ แต่เราอาศัยเทคนิคพิเศษเพียงนิดเดียว โดยการเพิ่มตัวอักษรแอส กี 8 นำหน้าชื่อ (caption) ของเมนูเท่านั้น ดังเช่น สมมติว่าผู้อ่านต้องการแสดงผลเมนู Help ชื่อ mnuHelp ชิดขอบขวาก็สามารถกระทำได้โดยการใส่ประโยคข้างล่างลงในโพรซีเดอร์ Form_Load ดังนี้

```
mnuHelp.Caption = Chr$(8) & mnuHelp.Caption
```

การใช้ตัวปฏิบัติการ Mod

ในการคำนวณค่าตัวนับในวงลูป ที่มีการรีเซตค่าทุกครั้งที่ค่าตัวนับเกินค่าสูงสุดที่ตั้งเอาไว้ โปรแกรมเมอร์มืออาชีพมักจะใช้ตัวปฏิบัติการ Mod แทนการใช้ประโยค If...Then...Else ดังตัวอย่างต่อไปนี้

```
Cnt% = Cnt% Mod MaxCnt% + 1
```

แทนการคำนวณด้วยโค้ดต่อไปนี้

```
Cnt% = Cnt% + 1
If Cnt% > MaxCnt% Then
   Cnt% = 1
End If
```

หรือ

```
If Cnt% > MaxCnt% Then
   Cnt% = 1
Else
   Cnt% = Cnt% + 1
End If
```

การเปลี่ยน wall paper ด้วย VB/Win

โดยปกติเราจะเปลี่ยน wall paper ของวินโดวส์โดยใช้ Control Panel แต่เราสามารถใช้ฟังก์ชันวินโดวส์ API ที่ชื่อ SystemParametersInfo เพื่อเปลี่ยน wall paper ตามไฟล์ .BMP ที่เรากำหนดให้ เช่นในตัวอย่างต่อไปนี้กำหนดให้ wall paper ตามไฟล์ 256color.bmp

```
Declare Function SystemParametersInfo Lib
   "User" (ByVal uAction As Integer,
   ByVal uParam As Integer, ByVal lpvParam
   As Any, ByVal fuWinIni As Integer) As
   Integer

Sub Form_Click ()
Const SPI_SETDESKWALLPAPER = 20
Const SPIF_UPDATEINIFILE = &H1
Const SPIF_SENDWININICHANGE = &H2
   Dim Dummy%

   Dummy% = SystemParametersInfo
   (SPI_SETDESKWALLPAPER, 0,
   "c:\windows\macha.bmp",
   SPIF_SENDWININICHANGE)
End Sub
```

หมายเหตุ ถ้าหากผู้อ่านต้องการให้มีการแก้ไขชื่อไฟล์ .BMP ที่ถูกกำหนดให้เป็น wall paper ลงในไฟล์ win.ini ผู้อ่านต้องกำหนดให้พารามิเตอร์ fuWinIni มีค่าเท่ากับ SPIF_UPDATEINIFILE Or SPIF_SENDWININICHANGE

ไดอะล็อกบ็อกซ์แบบ 3 มิติ

VB/Win 3.0 ไม่มีคำสั่งที่ช่วยให้การแสดงผลของไดอะล็อก-บ็อกซ์ เช่น message box , input box เป็นต้น แต่ผู้อ่านสามารถใช้ฟังก์ชันวินโดวส์ API ในไฟล์ไลบรารี CTL3D.DLL ช่วยในการแสดงผลแบบ 3 มิติได้ โดยอาศัย 3 ฟังก์ชันดังนี้

```
Declare Function Ctl3dRegister Lib
   "CTL3D.DLL" (ByVal hInstance As Integer)
   As Integer
Declare Function Ctl3dAutoSubClass Lib
   "CTL3D.DLL" (ByVal hInstance As Integer)
   As Integer
Declare Function Ctl3dUnregister Lib
   "CTL3D.DLL" (ByVal hInstance As Integer)
   As Integer
```

โดยมีขั้นตอนที่สำคัญอยู่ 2 ขั้นตอนดังนี้

1.ในโพรซีเดอร์ Form_Load ให้ใส่ประโยคต่อไปนี้ ลงในฟอร์มเริ่มต้น (startup form) เพื่อทำการลงทะเบียนแอพพลิเคชันที่ต้องการแสดงผลแบบ 3 มิติ โดยในที่นี้สมมติว่าฟอร์ม Form1 เป็นฟอร์มเริ่มต้น

```
Dummy% = Ctl3dRegister(Form1.hWnd)
Dummy% = Ctl3dAutoSubClass(Form1.hWnd)
```

2. ในโพรซีเดอร์ Form_Unload ให้ใส่ประโยคต่อไปนี้เพื่อยกเลิกการลงทะเบียนแอพพลิเคชันที่ต้องการแสดงผลแบบ 3 มิติ

```
Dummy% = Ctl3dUnregister(Form1.hWnd)
```

หมายเหตุ ถ้าหากโปรแกรมหยุดการทำงานด้วยประโยค End ก็จะทำให้เกิดข้อผิดพลาด GPF และ VB/Win จะถูกยกเลิก เพราะว่าการหยุดการทำงานของโปรแกรมด้วยประโยค End ไม่ทำให้เกิดเหตุการณ์ Form_Unload

การอ่านสถานะคีย์บอร์ดไทย/อังกฤษ

ผู้อ่านสามารถใช้ฟังก์ชัน Thai API ที่ชื่อ GetKeyboardLanguage ช่วยในการอ่านสถานะภาษาไทย/อังกฤษปัจจุบันของคีย์บอร์ดได้ ดังนี้

```
Declare Function GetKeyboardLanguage Lib
    "User" () As Integer

Sub Form_Click ()
Const KB_ENG = &H0
Const KB_THAI = &H1

    If GetKeyboardLanguage() = KB_ENG Then
       Print "English"
     Else
       Print "Thai"
     End If
End Sub
```

ตัวอย่างข้างต้นจะแสดงสถานะของคีย์บอร์ดทุกครั้งที่มีการคลิ๊กภายในฟอร์ม

VB/Win ค้นหาฟังก์ชัน API ไม่พบ

ผู้เขียนได้ทดลองใช้โปรแกรม API Text Viewer ที่มากับ VB/Win 4.0 คัดลอกฟังก์ชันวินโดวส์ API ที่ชื่อ GetWindowRect ปรากฏว่าเมื่อทดสอบโปรแกรม VB/Win แจ้งข้อผิดพลาดดังนี้ "Sub Or Function not defined" ตรงบรรทัดที่เราเรียกใช้ฟังก์ชันดังกล่าวเป็นครั้งแรก ทั้งนี้เนื่องจากประโยคการประกาศฟังก์ชันดังกล่าว ที่มากับไฟล์ WIN31API.TXT ผิดพลาดดังนี้

```
Declare Sub GetWindowRect Lib "GDI
  (ByVal hWnd As Integer, lpRect As . .y)
```

ซึ่งที่ถูกจะต้องแก้ไขใหม่ดังนี้

```
Declare Sub GetWindowRect Lib "USER"
  (ByVal hWnd As Integer, lpRect As Any)
```

การสร้างกราฟิกเคลื่อนไหว บน minimized form

ในขณะที่ฟอร์ม minimized จะปรากฏเป็นไอคอนนั้น ผู้อ่านสามารถสร้างภาพกราฟิกเคลื่อนไหว (animation) ภายในขอบเขตของไอคอนได้โดยใช้เทคนิคดังต่อไปนี้

1. กำหนดให้คุณสมบัติ Icon ของฟอร์มที่ต้องการมีค่าเท่ากับ None ในขณะออกแบบจากหน้าต่างคุณสมบัติ หรือโดยการเขียนโค้ดลงในโพรซีเดอร์เหตุการณ์ Form_Load จัดการในขณะรันก็ได้ดังนี้

```
Icon = LoadPicture()
```

2. ใส่โค้ดที่สร้างภาพกราฟิกเคลื่อนไหวไว้ในโพรซีเดอร์เหตุการณ์ Timer ของคอนโทรล Timer

แต่ในตัวอย่างนี้ผู้เขียนกำหนดให้วาดจุดแบบสุ่มลงในฟอร์ม minimized โดยใช้การวนลูปในโพรซีเดอร์เหตุการณ์ Form_Activate

```
Sub   Form_Activate ()
      Dim Cnt%

      Randomize Timer
      Do
         DoEvents
         PSet (Rnd * ScaleWidth, Rnd *
            ScaleHeight), Rnd * 15
         Cnt% = (Cnt% + 1) Mod 2000
         If Cnt% = 0 Then Cls
      Loop
End Sub
```

การสร้าง floating form

ผู้อ่านที่เคยใช้โปรแกรม corel draw จะเห็นว่ามีการสร้าง floating window มากมาย ซึ่งวินโดวส์เหล่านี้จะอยู่ด้านบนวินโดวส์อื่น ๆ ตลอดเวลา ผู้อ่านสามารถกำหนดให้ฟอร์มเป็นเช่นนี้ได้เช่นกัน โดยการใช้ฟังก์ชันวินโดวส์ API ที่ชื่อ SetWindowPos ดังนี้

```
Declare Function SetWindowPos Lib "user"
  (ByVal h%, ByVal hb%, ByVal x%,
  ByVal y%, ByVal cx%, ByVal cy%,
  ByVal f%) As Integer

Sub   FloatWindow (frm As Form, FloatFlag%)
      Const SWP_NOMOVE = 2
      Const SWP_NOSIZE = 1
      Const HWND_TOPMOST = -1
      Const HWND_NOTOPMOST = -2

      Dim success%, Flag%

      Flag% = SWP_NOMOVE Or SWP_NOSIZE
      If FloatFlag% Then
        ' Floating form.
        success% = SetWindowPos(frm.hWnd,
        HWND_TOPMOST, 0, 0, 0, 0, Flag%)
      Else
        ' Unfloating form.
        success% = SetWindowPos(frm.hWnd,
        HWND_NOTOPMOST, 0, 0, 0, 0, Flag%)
      End If
End Sub
```

การคัดลอกไฟล์บีบอัดขนาด

ผู้อ่านสามารถคัดลอกไฟล์ที่ถูกบีบอัดด้วยโปรแกรม COMPRESS.EXE ของวินโดวส์ได้ด้วยฟังก์ชันวินโดวส์ API ที่ชื่อ LZCopy ซึ่งฟังก์ชันดังกล่าวจะตรวจสอบไฟล์ที่จะคัดลอก โดยถ้าหากไฟล์ดังกล่าวเป็นไฟล์บีบอัด ฟังก์ชันก็จะทำการคัดลอกพร้อมกับขยายขนาดของไฟล์โดยอัตโนมัติ แต่ถ้าหากไฟล์ต้นทางเป็นไฟล์ปกติฟังก์ชันก็จะทำการคัดลอกไฟล์แบบธรรมดา

```
Declare Function LZCopy Lib "LZexpand.dll"
  (ByVal hfSource As Integer,
  ByVal hfDest As Integer) As Long
```

→

```
Function CopyByLZCopy& (srcfile$, dstname$)
  Dim CodeFlag&, SrcFileNum%, DstFileNum%
  Dim SrcHdl%, DstHdl%

  SrcFileNum% = FreeFile
  Open srcfile$ For Input As #SrcFileNum%
  DstFileNum% = FreeFile
  Open dstname$ For Output As
       #DstFileNum%
  SrcHdl% = FileAttr(SrcFileNum%, 2)
  DstHdl% = FileAttr(DstFileNum%, 2)

  CopyByLZCopy& =  LZCopy(SrcHdl%,
                   DstHdl%)

       Close SrcFileNum%, DstFileNum%
End Function
```

ในตัวอย่างข้างต้นสมมติว่าผู้อ่านต้องการคัดลอกไฟล์ต้นทาง a:\setup1.ex_ ไปยังไฟล์ปลายทางที่ชื่อ c:\windows\setup1.exe ก็สามารถเรียกใช้ฟังก์ชัน CopyByLZCopy& ได้ดังนี้

```
RCode& = CopyByLZCopy& ("a:\setup1.ex_",
\windows\setup1.exe")
```

หมายเหตุ ถ้าหากเกิดข้อผิดพลาดฟังก์ชัน LZCopy จะส่งค่ากลับที่น้อยกว่า 0 เสมอ

การบีบอัดไฟล์อย่างปลอดภัย ด้วย COMPRESS.EXE

โปรแกรมบีบอัดข้อมูล COMPRESS.EXE ที่มากับ VB/Win Prof Edition จะถูกใช้ในการบีบอัดไฟล์สำหรับสร้างชุด setup ข้อผิดพลาดประการหนึ่งที่โปรแกรมเมอร์พบบ่อยมากก็คือ เมื่อผู้อ่านบีบไฟล์ setup1.exe ดังนี้

```
compress setup1.exe setup1.ex_
```

ปรากฏว่าเมื่อทำการ setup โปรแกรมโหลดเดอร์ setup.exe จะแจ้งข้อผิดพลาด Cound not execute: SETUP1.EX ทุกครั้ง และทำการการติดตั้งไม่สำเร็จ ซึ่งในกรณีนี้ไมโครซอฟท์แนะนำให้ทำการบีบอัดไฟล์ setup1.exe ด้วยพารามิเตอร์ -r ดังนี้

```
compress -r setup1.exe
```

ซึ่งไฟล์ setup1.ex_ ที่ได้จะสามารถนำไปใช้ได้อย่างไม่มีปัญหา

ค้นหาฐานข้อมูลด้วย SQL

งานชนิดหนึ่งที่ไม่สามารถหลีกเลี่ยงได้ในการเขียนโปรแกรมจัดการฐานข้อมูล นั่นก็คือการค้นหาข้อมูล ซึ่ง database JET engine ได้เตรียมทางเลือกมากมายในการค้นหาข้อมูล เช่น ใช้วิธี Find, Seek หรือ Filter เป็นต้น แต่ถ้าหากฐานข้อมูลมีขนาดใหญ่ ผู้เขียนขอแนะนำให้ผู้อ่านใช้ SQL ในการค้นหาข้อมูล เพราะสามารถช่วยลดเวลาในการค้นหาได้มากทีเดียว โดยเฉพาะ JET 2.5/3.0 ที่ใช้เทคโนโลยี Rushmore ซึ่งสามารถทำการออปติไมซ์ประโยค SQL โดยอัตโนมัติได้อีกด้วย สำหรับประโยคที่ใช้ในการค้นหาข้อมูลโดยทั่วไปจะมีรูปแบบดังนี้

```
SELECT field1, field2,...
FROM table1, table2,...
WHERE expr1, expr2,...
ORDER BY field1, field2,...
```

โดยที่ fieldN หมายถึง ฟิลด์
tableN หมายถึง ตารางฐานข้อมูล
exprN หมายถึง นิพจน์เปรียบเทียบ

เทคนิคการหน่วงเวลา โดยไม่ใช้คอนโทรล Timer

โดยปกติเราจะใช้คอนโทรล Timer ของ VB/Win ในการหน่วงเวลาตามที่กำหนด ซึ่งผู้อ่านจะเห็นว่าในกรณีที่เราไม่ต้องการใช้ความสามารถอื่น ๆ ของคอนโทรล Timer ก็จะทำให้โปรแกรมมีขนาดโตขึ้นโดยไม่จำเป็น ดังนั้นผู้เขียนจึงขอแนะนำรูทีน DelayTime ข้างล่าง ซึ่งผู้อ่านสามารถนำไปใช้หน่วงเวลาได้ตามที่ต้องการ

```
Sub   DelayTime (DelayTime!)
      Dim LoopTime!

      Const SecInDay = 24& * 60& * 60&
      LoopTime! = Timer + DelayTime!
      If LoopTime! > SecInDay Then
            LoopTime! = LoopTime! - SecInDay
            Do While Timer > LoopTime!
                  DoEvents
            Loop
      End If
      Do While Timer < LoopTime!
            DoEvents
      Loop
End Sub
```

DAO ของ VB/Win 4.0

VB/Win 4.0 มาพร้อมกับ database JET engine 2.5 (16/32 บิต) และ 3.0 (32 บิต) ซึ่งไมโครซอฟท์แนะนำให้ใช้วิธีใหม่แทนวิธีเก่าที่ใช้กับ JET 1.1 ซึ่งมากับ VB/Win 3.0 ดังรายการต่อไปนี้

JET 1.1	JET 2.5/3.0
CreateDatabase statemant	**CreateDatabase** method
CompactDatabase statemant	**CompactDatabase** method
Snapshot object	Snapshot-type
Dynaset object	Dynaset-type
Table object	Table-type
CreateDynaset method	**OpenRecordset** with **dbOpenDynaset** option
CreateSnapshot method	**OpenRecordset** with **dbOpenSnapshot** option

→

OpenTable method	OpenRecordset with dbOpenTable option
BeginTrans	Workspace.BeginTrans
CommitTrans	Workspace.CommitTrans
Roolback	Workspace.Rollback
DeleteQueryDef	QueryDefs.Delete
ListFields	Fields collection
ListTables	TableDefs collection
ListParameters	Parameters collection
OpenQueryDef	QueryDefs collection

15 ข้อแนะนำในการใช้ JET

บรรดาโปรแกรมเมอร์มืออาชีพที่ใช้ VB/Win พัฒนาโปรแกรมจัดการฐานข้อมูลด้วย JET มีข้อแนะนำ 15 ประการ ในการเพิ่มความเร็วให้กับโปรแกรม ดังต่อไปนี้

1. ใช้วิธี Seek แทนวิธี Find ในการค้นหาข้อมูล
2. ใช้ table แทน dynaset หรือ shapshot
3. กำหนด index หรือใช้ SQL แทนวิธี Sort
4. แทนการใช้วิธี Filter ด้วย Seek หรือ SQL
5. ใช้ bookmark แทนวิธี Find
6. ใช้วิธี ReQuery แทนการเปิด table อีกครั้ง
7. สร้าง static SQL แทนการใช้ dynamic SQL
8. แทนที่ตัวเลข floating-point ด้วยเลขจำนวนเต็ม
9. ใช้ Clone กับ dynaset แทนการเปิด table ใหม่
10. ไม่ควรสร้างตารางในไฟล์ .MDB มากเกินไป
11. ประมวลผลข้อมูลจำนวนมากแบบ transaction
12. แทนที่ข้อมูล variant ด้วย string
13. สร้างตัวแปรแทนออบเจ็กต์ แทนการอ้างอิงด้วยชื่อออบเจ็กต์ที่ยาวมาก ๆ
14. เปลี่ยนจากวิธี List ต่าง ๆ ด้วย collection
15. ใช้วิธี Refresh กับ collection เท่าที่จำเป็น

วิธีแปลงไฟล์ไอคอนเป็นไฟล์บิตแม็พ

ผู้อ่านสามารถแปลงไฟล์ไอคอน (.ICO) ให้เป็นไฟล์บิตแม็พ (.BMP) ได้โดยใช้คำสั่งของ VB/Win และคอนโทรล picture box ดังในรูทีนตัวอย่างข้างล่าง ConvertICO2BMP จะแปลงไฟล์ไอคอนเป็นไฟล์บิตแม็พที่มีชื่อและไดเรคทอรีเดียวกันแต่มีนามสกุลเป็น .BMP เท่านั้น

```
Sub   ConvertICO2BMP (iconfilename$)
      Dim i%, bmpfilename$

      Picture1.AutoSize = True
      Picture1.Picture = LoadPicture
                         (iconfilename$)

      i% = InStr(iconfilename$, ".")
      bmpfilename$ = Left$(iconfilename$, i%)
                     & "bmp"
      SavePicture Picture1.Image,
                  bmpfilename$
End Sub
```

การใช้ข้อมูลชนิด Date ใน SQL

เนื่องจาก VB/Win สนับสนุนรูปแบบของวันที่ในหลาย ๆ มาตรฐาน เช่น mm/dd/yyyy หรือ dd/mm/yyyy เป็นต้น ซึ่งผู้อ่านสามารถตั้งรูปแบบได้จาก CONTROL PANEL ดังนั้นในการใช้ข้อมูลชนิด Date ใน SQL จึงต้องคำนึงถึงจุดนี้ด้วย โดยผู้เขียนขอแนะนำให้แปลงข้อมูลชนิด Date ด้วยฟังก์ชัน Format เสียก่อนจากนั้นจึงนำมาใช้ใน SQL และที่สำคัญต้องล้อมรอบวันที่ด้วยเครื่องหมาย # เท่านั้น ดังโค้ดข้างล่างเป็นตัวอย่างการค้นหารหัสพนักงาน (EmployeeID) ที่เกิดในวันที่ bdate จากตารางที่ชื่อ Personnel โดยในตัวอย่างจะทำการแปลงข้อมูล bdate ให้เป็นข้อมูลสตริง strdate$ เสียก่อน

```
Set Db = OpenDatabase(filename$)
strdate$ = Format$(bdate, "short date")
msgSQL$ = "SELECT EmployeeID FROM
          Personnel WHERE BirthDate=#"
          & strdate$ & "# ORDER BY
          EmployeeID"
Set Ss = Db.CreateSnapshot(msgSQL$)
```

ทำไมฟังก์ชัน GetVersion รายงานเวอร์ชันของวินโดวส์ 3.11 เป็น 3.10

ผู้อ่านที่เคยใช้ฟังก์ชันวินโดวส์ API ที่ชื่อ GetVersion ในการอ่านเวอร์ชันของวินโดวส์ จะพบว่าภายใต้วินโดวส์ 3.11 หรือ 3.11 for workgroup ฟังก์ชันนี้จะรายงานเป็นเวอร์ชัน 3.10 ทั้งนี้เนื่องจากไมโครซอฟท์กำหนดให้ฟังก์ชันดังกล่าวยังคงรายงานเวอร์ชันเป็น 3.10 เพราะมีแอพพลิเคชันมากมายที่ตรวจสอบเวอร์ชันของวินโดวส์ด้วยฟังก์ชัน GetVersion และทำงานภายใต้เวอร์ชัน 3.10 เท่านั้น

การอ่านเวอร์ชันของวินโดวส์ที่ถูกต้องจากไฟล์ USER.EXE

ผู้อ่านสามารถอ่านเวอร์ชันที่ถูกต้องของวินโดวส์ได้จากไฟล์ USER.EXE โดยใช้ฟังก์ชันวินโดวส์ API ที่ชื่อ GetFileVersionInfo โดยในตัวอย่างนี้ฟังก์ชัน GetWinVersion! จะรายงานหมายเลขเวอร์ชันเป็นข้อมูลเลขจำนวนจริง single ดังนี้

```
Declare Function GetFileVersionInfo Lib
   "VER.DLL" (ByVal lpszFileName As String,
   ByVal handle As Any, ByVal cbBuf As
   Long, ByVal lpvData As String)
   As Integer

Function GetWinVersion! ()
   Dim i%, Dummy%
   Dim buffer As String * 255

   Dummy% = GetFileVersionInfo("USER.EXE",
         , 254, buffer)
   i% = InStr(1, buffer, "FileVersion")
   GetWinVersion! = Mid(buffer, i%+ 12, 4)
End Function
```

ตรวจสอบการติดตั้งซาวนด์การ์ด

ผู้อ่านที่เคยอ่านบทความเปิดโลกมัลติมีเดียของผู้เขียน ก็คงทราบวิธีการเล่นไฟล์ .WAV มาแล้วพอสมควร แต่คงยังไม่ทราบวิธีการตรวจสอบการติดตั้งซาวนด์การ์ดบนเครื่อง ซึ่งเราสามารถตรวจสอบ ๆ ได้โดยใช้ฟังก์ชันในกลุ่มของ Multimedia Low-Level Waveform Audio Services ชื่อ waveOutGetNumDevs ดังตัวอย่างข้างล่าง

```
Declare Function waveOutGetNumDevs Lib
            "MMSYSTEM" () As Integer

Sub Form_Load ()
      If waveOutGetNumDevs() > 0 Then
        MsgBox "Sound card installed."
      Else
        MsgBox "No Sound card install."
      End If
End Sub
```

ค้นหาไดรฟ์ว่างของระบบเครือข่าย

ในระบบเครือข่าย windows for workgroup สนับสนุนไดรฟ์ตั้งแต่ A: ถึง Z: ซึ่งการแม็พไดรฟ์ในระบบเครือข่ายสามารถที่จะกระทำโดยไม่จำเป็นต้องเรียงลำดับไดรฟ์อย่างต่อเนื่องก็ได้ ดังนั้นในการค้นหาไดรฟ์ว่างในระบบเครือข่ายจึงเป็นสิ่งที่จำเป็น ซึ่งผู้อ่านสามารถค้นหาไดรฟ์ว่างในระบบเครือข่ายได้โดยใช้ฟังก์ชันวินโดวส์ API ที่ชื่อ GetDriveType โดยในตัวอย่างต่อไปนี้ฟังก์ชัน NextFreeDrive จะรายงานชื่อไดรฟ์ว่างในรูปแบบ เช่น E:, F: หรือ G: เป็นต้น

```
Declare Function GetDriveType Lib "kernel"
  (ByVal nDrive As Integer) As Integer

Function NextFreeDrive () As String
      Dim DrvNum%

      DrvNum% = -1
      Do
            ' Start at zero.
          DrvNum% = DrvNum% + 1
      Loop Until GetDriveType(DrvNum%) = 0

      NextFreeDrive = Chr(DrvNum% + 65) & ":"
End Function
```

หมายเหตุ พารามิเตอร์ nDrive จะเริ่มต้นด้วยค่าเท่ากับ 0 หมายถึงไดรฟ์ A:, 1=B:, 3=C: ตามลำดับ

แสดงทูลบาร์ของ Microsoft Word ด้วย OLE 2.0

OLE เป็นสิ่งหนึ่งที่โปรแกรมเมอร์ในบ้านเราไม่ค่อยคุ้นเคย แต่สำหรับโปรแกรมเมอร์ในสหรัฐอเมริกานั้นมีการโปรแกรมโดยใช้ OLE กันอย่างแพร่หลาย และเทคนิคอย่างหนึ่งที่ใช้กันบ่อยก็คือ การแสดงผลเฉพาะทูลบาร์ของ Microsoft Word ด้วยคอนโทรล OLE 2.0 ของ VB/Win ซึ่งผู้อ่านสามารถกระทำได้โดยเดินตามลำดับดังนี้

1. เพิ่มคอนโทรล OLE 2.0 ลงในฟอร์ม

2. ในโหมดออกแบบใช้คำสั่ง Insert Object แทรก Microsoft Word 6.0 ลงในคอนโทรล OLE 2.0

3. เขียนโค้ดต่อไปนี้เพื่อสร้างออบเจ็กต์ Word Basic ของ Microsoft Word 6.0 ดังนี้

```
Sub   Form_DblClick ()
      Dim ObjWordBasic As Object

      Ole1.Action = 7
      Set ObjWordBasic =
            CreateObject("Word.basic")
      ObjWordBasic.ViewToolbars
            "Standard", , , , , , , 1
End Sub
```

หมายเหตุ ในการรันตัวอย่างนี้ เมื่อผู้อ่านดับเบิ้ลคลิ๊ก บนฟอร์ม ก็จะปรากฏทูลบาร์ของ Microsoft Word ขึ้นมา ซึ่งผู้อ่านสามารถเคลียร์ได้โดยการกดคีย์ Esc หรือเลือกแสดงทูลบาร์อื่น ๆ โดยการคลิ๊กปุ่มขวามือของเม้าส์ภายในทูลบาร์ ก็จะปรากฏเมนูแบบ popup ให้เลือก

การใช้ SpellChecker ของ Microsoft 6.0 ด้วย VB/Win

ในการเขียนโปรแกรมชนิดตรวจคำผิด (spell checker) นั้นไม่ใช่เรื่องง่าย ๆ ดังนั้นถ้าหากผู้อ่านใช้ Microsoft Word 6.0 ผู้อ่านก็สามารถยืมคุณสมบัติ SpellChecker มาใช้ตรวจสอบคำผิดของสตริงของ VB/Win ได้ โดยอาศัยความสามารถด้าน OLE Automation ของ OLE 2.0 ดังในตัวอย่างต่อไปนี้จะใช้ SpellChecker ตรวจสอบคำผิดในคอนโทรล text box เมื่อผู้อ่านคลิ๊กที่ปุ่มคำสั่ง Command1 ซึ่งขั้นตอนการสร้างตัวอย่างมีดังนี้

1. เพิ่มคอนโทรล OLE 2.0 ลงในฟอร์ม

2. ในโหมดออกแบบใช้คำสั่ง Insert Object แทรก Microsoft Word 6.0 ลงในคอนโทรล OLE 2.0

3. เพิ่มคอนโทรล Command button ลงในฟอร์มและกำหนดชื่อ Command1 ให้คุณสมบัติ Caption เท่ากับ "SpellChecker"

4. เพิ่มคอนโทรล text box ลงในฟอร์มและ กำหนดชื่อ Text1 ให้คุณสมบัติ MultiLine มีค่าเท่ากับ True และคุณสมบัติ Text เท่ากับ "Hellp Visual BASIC nice readr."

5. เขียนโค้ดต่อไปนี้เพื่อสร้างออบเจ็กต์ Word Basic และเรียกใช้ SpellChecker ของ Microsoft Word 6.0 เพื่อตรวจสอบคำใน Text1 ลงในโพรซีเดอร์เหตุการณ์ Command1_Click ดังนี้

```
Sub Command1_Click ()
  Dim ObjWordBasic As Object, p$

  Set ObjWordBasic =
      CreateObject("Word.Basic")
  ObjWordBasic.FileNew
  ObjWordBasic.Insert Text1.Text

  On Error Resume Next

  ObjWordBasic.ToolsSpelling
  ObjWordBasic.EditSelectAll
  ObjWordBasic.SetDocumentVar "VBvar",
  ObjWordBasic.Selection
  p$ = ObjWordBasic.GetDocumentVar("VBvar")
  Text1.Text = Left(p$, Len(p$) - 1)
End Sub
```

จากนั้นผู้อ่านสามารถทดสอบการใช้ SpellChecker ได้โดยการคลิ๊กที่ปุ่มคำสั่ง Command1 เท่านั้น

หมายเหตุ ถ้าหากคุณสมบัติ MultiLine ของคอนโทรล text box เท่ากับ False VB/Win จะค้างทันที และสามารถส่งผลต่อส่วนอื่น ๆ ของวินโดวส์ด้วย

การอ่านสถานะฟอร์มของ VB/Win

ผู้อ่านจะเห็นว่าเราสามารถกำหนดสถานะการแสดงผลของฟอร์มให้เป็นแบบ modal หรือ modeless ได้โดยการกำหนดค่า 1 หรือ 0 ตามลำดับให้กับด้วยวิธี Show แต่เราไม่สามารถอ่านสถานะของฟอร์มได้เลยด้วยคำสั่งของVB/Win แต่ในการเขียนโปรแกรมผู้เขียนอ่านสถานะของฟอร์มด้วยฟังก์ชันวินโดวส์ API ที่ชื่อ GetWindowLong โดยในฟังก์ชันตัวอย่าง GetFormState จะรายงานสถานะของฟอร์มที่มีหมายเลข handle เท่ากับ frmhWnd% โดยจะรายงานค่า 1 สำหรับฟอร์ม modal และ 0 สำหรับฟอร์ม modeless

```
Declare Function GetWindowLong Lib "user"
  (ByVal hWnd As Integer,
  ByVal nIndex As Integer) As Long

Const GWL_STYLE = (-16)
Const WS_DISABLED = &H8000000

Function GetFormState (frmhWnd%) As Integer
  Dim WinStyle&

  WinStyle& = GetWindowLong(frmhWnd%,
            GWL_STYLE)
  If WinStyle& And WS_DISABLED Then
       GetFormState = 1    ' modal
  Else
       GetFormState = 0    ' modeless
  End If
End Function
```

การจำกัดตัวอักษรใน combo box

VB/Win ไม่สนับสนุนวิธีการจำกัดจำนวนตัวอักษรที่สามารถพิมพ์ลงใน text box ของคอนโทรล combo box ที่คุณสมบัติ Style มีค่าเท่ากับ 0 (dropdown) แต่ผู้อ่านสามารถจำกัดจำนวนตัวอักษรได้โดยใช้ฟังก์ชันวินโดวส์ API ที่ชื่อ SendMessage ดังตัวอย่างข้างล่างจำกัดการพิมพ์ที่ 9 ตัวอักษร

```
Declare Function SendMessage Lib "User"
  (ByVal hWnd As Integer, ByVal wMsg As
  Integer, ByVal wParam As Integer,
  lParam As Any) As Long

Const WM_USER = &H400
Const CB_LIMITTEXT = WM_USER + 1

Sub Form_Load ()
  Dim Dummy%, TxtLen%

  Show
  TxtLen% = 9
  Dummy% =   SendMessage(Combo1.hWnd,
             CB_LIMITTEXT, TxtLen%, 0)
End Sub
```

การสร้างฟอร์มที่ไม่มี title bar

เมื่อผู้อ่านโหลด VB/Win ก็จะเห็นว่าโปรแกรมจะแสดงฟอร์มเริ่มต้น (startup form) เพื่อแสดงรายละเอียดของ VB/Win ซึ่งเป็นฟอร์มที่ไม่มี title bar และสำหรับ VB/Win ผู้อ่านก็สามารถสร้างฟอร์มดังกล่าวได้เช่นกัน โดยขั้นตอนดังต่อไปนี้

1. กำหนดให้คุณสมบัติ Caption ของฟอร์มเท่ากับสตริงว่าง ("")

2. กำหนดให้คุณสมบัติดังต่อไปนี้ ControlBox, MaxButton และ MaxButton มีค่าเท่ากับ False

เมื่อผู้อ่านรันฟอร์มดังกล่าวก็จะเป็น ๆ ฟอร์มที่ไม่มี title bar

หมายเหตุ ผู้อ่านไม่สามารถเลื่อนฟอร์มไป-มาโดยการใช้เม้าส์หรือคีย์บอร์ด

และสำหรับผู้อ่านที่ต้องการปรึกษาหรือสอบถามปัญหา หรือข่าวคราวต่าง ๆ ของ VB/Win ก็สามารถเขียนจดหมายมาพูดคุยกันได้ตามที่อยู่ดังนี้

ตู้ ป.ณ. 35 ปทฝ.ดินแดง
เขตห้วยขวาง
กรุงเทพฯ 10407